DETAILED ACTION

1. This action is in response to Applicant's submission filed 7/24/09, responding to the 8/8/09 Office action which detailed the rejection of claims 7-9, 14-18, and 24-30. Claims 7, 9, 15-18, 26 have been amended, claims 14, 27-28 have been canceled, and new claims 31-47 have been added. Claims 7-9, 15-18, 24-26, and 31-47 remain pending in the application and have been fully considered by the examiner.

Response to Amendment/Arguments

2. Applicant's arguments with respect to the rejection under 35 U.S.C. § 112, first paragraph, in combination with the claim amendments, are persuasive. Therefore, the rejection is withdrawn.

3. Applicant's arguments on pages 21-28 filed 5/15/09 with respect to the rejections under 35 U.S.C. § 103(a) filed 5/15/09 are moot in view of the new grounds of rejection below.

Claim Objections

4. Claim 39 is objected to because of the following informalities: Claim 39 recites: "...wherein application program is written in the JAVA programming language the step of detecting..." This should instead read: "...wherein said application program is written in the JAVA programming language and the step of detecting..." Appropriate correction is required.

5. Claim 47 is objected to under 37 CFR 1.75(c), as being of improper dependent form for failing to further limit the subject matter of a previous claim. Applicant is required to cancel the

claim(s), or amend the claim(s) to place the claim(s) in proper dependent form, or rewrite the claim(s) in independent form. Claim 47 reiterates the same limitations of parent claim 46.

Claim Rejections - 35 USC § 112

6. The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

7. Claims 33, 41 and 44 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention.

8. Claim 33 recites: "the processing of the alert thread is only interrupted momentarily." Applicant has not pointed out where the new claim is supported, nor does there appear to be a written description of this limitation in the application as filed. The closest disclosure is found on page 7 lines 21-23 of the specification: "This is an operation which can be carried out quickly and thus the processing of the "altered" thread is only interrupted momentarily before the thread resumes processing." The limitation will be interpreted with respect to the disclosure on page 7 related to the "altered" thread.

9. Claim 41 recites: "...wherein a multicast socket is used for a distributed run time..." in lines 1-2. Applicant has not pointed out where the new claim is supported, nor does there appear to be a written description of this limitation in the application as filed. No disclosure was found

regarding multicast sockets. This limitation will be broadly interpreted according to a general network as described on page 2 lines 19-21.

10. Claim 44 recites: "...writing the value from the network packet..." in lines 1-2. Applicant has not pointed out where the new claim is supported, nor does there appear to be a written description of this limitation in the application as filed. No disclosure was found regarding network packets. This limitation will be broadly interpreted according to a general network communication as described on page 2 lines 19-21.

11. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

12. Claim 33, 37 and 38 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

13. Claim 33 recites the limitation "the notification" in line 1. There is insufficient antecedent basis for this limitation in the claim. For the purpose of further examination, this claim will be interpreted as dependent upon claim 32.

14. Claim 33 recites the limitation "the alert thread" in line 2. There is insufficient antecedent basis for this limitation in the claim. As noted above, this limitation has issues regarding the written description requirement. For the purpose of further examination, this limitation will be interpreted as "a thread that has been altered" as described on page 7 lines 21-23 of the specification.

15. Claim 33 recites the limitation "the other thread" in line 2. There is insufficient antecedent basis for this limitation in the claim. For the purpose of further examination, this limitation will be interpreted as "another thread."

16. Claims 37 and 38 contain the trademark/trade name "JAVA". Where a trademark or trade name is used in a claim as a limitation to identify or describe a particular material or product, the claim does not comply with the requirements of 35 U.S.C. 112, second paragraph. See Ex parte Simpson, 218 USPQ 1020 (Bd. App. 1982). The claim scope is uncertain since the trademark or trade name cannot be used properly to identify any particular material or product. A trademark or trade name is used to identify a source of goods, and not the goods themselves. Thus, a trademark or trade name does not identify or describe the goods associated with the trademark or trade name. In the present case, the trademark/trade name is used to identify/describe the JAVA programming language and, accordingly, the identification/description is indefinite. The claims should be amended to recite "JAVA programming language."

Claim Rejections - 35 USC § 102

17. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

18. Claims 7, 8, and 24 are rejected under 35 U.S.C. 102(a) as being anticipated by "JavaSplit: A Runtime for Execution of Monolithic Java Programs on Heterogeneous Collections of CommodityWorkstations" by Factor et al. ("Factor").

In regard to claim 7, Factor discloses:

In a multiple computer system including a plurality of single computers interconnected via a communications link, a method of loading an application program onto each of said plurality of single computers, the application program having application program code including a plurality of code threads all intended to execute on and reference a single computer having a single processing unit or symmetric multiple processing units and a single independent local memory with a local memory capacity that is not shared with any other single computer of said plurality of single computers, See at least page 1 right column, e.g.: "It is able to execute a pre-existing, multithreaded application on any given heterogeneous set of machines." Note that the system described in Factor utilizes the Java programming environment which is understood to execute first and foremost on a single processing unit with a single independent local memory. Factor's disclosure implies this by suggesting that their system can take such an application and execute it in a multiple computer system.

the method comprising the steps of:

loading the application program written to operate only on a single computer onto each different computer of said plurality of single computers; and See the top of the

left column on page 2: "In JavaSplilt, each node executes its part of the rewritten application on its local standard JVM."

modifying the application program on each said different computer before execution of said corresponding portion of the application program written to operate only on a single computer on each said different single computer; See section 4 on page 3, e.g. "Bytecode instrumentation."

different portions of said application program being simultaneously executable on each different one of the plurality of single computers with each different one of the plurality of single computers having a different independent local memory accessible only by a corresponding portion of the application program. See page 1, right column, e.g.: "When combined with the code of the runtime, the instrumented program becomes a distributed application, ready to be executed on multiple nodes (Figure 1)."

In regard to claim 8, the above rejection of claim 7 is incorporated. Factor further discloses: *wherein the step of modifying the application program is different for different computers*. See page 3, bottom of right column, e.g. "ships the thread to a node chosen by the load balancing function."

In regard to claim 24, the above rejection of claim 7 is incorporated. Factor further discloses: *wherein said program written to operate on only a single computer is a program written to execute within a local processor or processors and local memory coupled to the processor or processors within the single computer*. See at least page 1

right column, e.g.: "It is able to execute a pre-existing, multithreaded application on any given heterogeneous set of machines." Note that the system described in Factor utilizes the Java programming environment which is understood to execute first and foremost on a single processing unit with a single independent local memory. Factor's disclosure implies this by suggesting that their system can take such an application and execute it in a multiple computer system.

Claim Rejections - 35 USC § 103

19. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

20. Claim 9 is rejected under 35 U.S.C. 103(a) as being unpatentable over Factor in view of U.S. Patent No. 5,802,585 to Scales et al. (hereinafter "Scales").

In regard to claim 9, the above rejection of claim 7 or 8 is incorporated. Factor further discloses:

Factor does not expressly disclose the remaining limitations. However, Scales teaches:

(i) detecting instructions in the unmodified application program which reference the same common memory records; See Scales column 4:38-42, e.g. "coherency."

(ii) listing all such commonly referenced memory records and providing a naming tag for each said listed commonly referenced memory record, See column 6:20-21, e.g. "table," and column 11:6-10, e.g. "ID."

(iii) detecting those instructions which write to, or manipulate the contents of, any of said listed commonly referenced memory records, and See column 4:30-32, e.g. "stores."

(iv) generating an alert instruction following each said detected commonly referenced memory record write or manipulate instruction, said alert instruction forwarding the re-written or manipulated contents and name tag of each said re-written or manipulated listed memory record. See column 1:43-49, e.g. "message passing" and at least column 1 lines 50-57, e.g. "miss check."

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Scales' shared memory with Factor's loading in order to provide coherency (see Scales column 1:20-26)

21. Claims 15-18, 26, 31-32, 34-37, and 39-44 are rejected under 35 U.S.C. 103(a) as being unpatentable over Factor in view of Scales, and further in view of U.S. Patent 6574674 to May et al. ("May").

In regard to claim 31, Factor discloses:

A method of compiling or modifying an application program written to include a plurality of instruction code threads intended to execute on and reference only a single computer having a single central processing unit (CPU) or symmetric multiple processing units and a single independent local memory that is not shared with any other computer of said plurality of single computers but to run simultaneously on each one of a plurality of computers interconnectable via a communications link, with different portions of said application program being simultaneously executable on different ones of said plurality of single computers with each one of the plurality of single computers having the independent local memory accessible only by the corresponding portion of the application program, See at least page 1 right column, e.g.: "It is able to execute a pre-existing, multithreaded application on any given heterogeneous set of machines." Note that the system described in Factor utilizes the Java programming environment which is understood to execute first and foremost on a single processing unit with a single independent local memory. Factor's disclosure implies this by suggesting that their system can take such an application and execute it in a multiple computer system.

Factor does not expressly disclose the remaining limitations. However, Scales teaches:

(i) detecting instructions in the unmodified application program which reference the same common memory records; See Scales column 4:38-42, e.g. "coherency."

(ii) listing all such commonly referenced memory records and providing a naming tag for each said listed commonly referenced memory record; See column 6:20-21, e.g. "table," and column 11:6-10, e.g. "ID."

(iii) detecting those instructions which write to, or manipulate the contents of, any of said listed commonly referenced memory records; and See column 4:30-32, e.g. "stores."

Factor and Scales do not expressly disclose:

(iv) generating and inserting an alert instruction into the unmodified application program to create the modified application program for handling by a distributed run time (DRT) following each said detected commonly referenced memory record write or manipulate instruction indicating that the contents or value of the commonly referenced memory record were re-written or manipulated and may have changed during execution of a code thread, said alert instruction being operative for initiating propagation of the rewritten or manipulated contents and name tag of each said re-written or manipulated listed commonly referenced memory record via the communications link to the distributed run times (DRTs) of each other of the single computers. However, May teaches that an OM system (i.e. "DRT") is used to handle messages which propagate an indication of write of manipulate instructions of commonly referenced memory records. See column 14 line 65 - column 15 line 23, e.g.

> Each operator message specifies an action and specifies the object on which the action is to be performed. The operator messages include a clear message, a move message, an add message, a replace message, an update message, and a delete message.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Factor's distributed system with May's OM system in order to process operations of shared data as suggested by May.

In regard to claims 15-18, the above rejection of claim 31 is incorporated. Factor does not expressly disclose: *carried out: prior to loading the application program onto each said computer*; *during loading of the application program onto each said computer*; *by just-in-time compilation*; or *by re-compilation after loading*. However, Factor teaches modification of source and byte code. See page 1, right column. Factor also discloses just-in-time compilation. See abstract. The modification of source code is accomplished prior to loading. Finally, Factor discloses instrumentation after first loading the original code. See table 1 on page 6. It would have been obvious to one of ordinary skill in the art at the time the invention was made to try the various sequences source modification since Factor has identified each of these situations with respect to the opportunity for optimization.

In regard to claim 26, the above rejection of claim 31 is incorporated. Factor further discloses: *wherein said program written to operate on only a single computer is a program written to execute within a local processor or processors and local memory coupled to the processor or processors within the single computer*.

In regard to claim 32, the above rejection of claim31 is incorporated. Factor further discloses: *(i) directly notify and propagate* ... See section 3, page 3, middle of left column, e.g. "propagated from a writer to its home as a difference." Factor does not expressly disclose: *to all other DRTs executing on each other one of the plurality of single computers of the re-writing or manipulation and possible change of contents or*

value of the commonly referenced memory record and then resumes processing; and (ii) indirectly notify and propagate by instructing another thread to notify and propagate the all other DRTs executing on each other one of the plurality of single computers of the re-writing or manipulation and possible change of contents or value of the commonly referenced memory record and then resumes processing. However, May teaches that notification of manipulation of memory is propagated to DRTs. See column 14 line 65 - column 15 line 23, as cited above. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Factor's notification with May's DRTs in order to efficiently manage data as suggested by May (see column 2 lines 16-35).

In regard to claim 34, the above rejection of claim 31 is incorporated. Factor further discloses: *wherein the communication link comprises the Internet.* See section 2, bottom of page 2, right column.

In regard to claim 35, the above rejection of claim 31 is incorporated. Factor further discloses: *wherein the communication link comprises an intranet.* See section 2, bottom of page 2, right column.

In regard to claim 36, the above rejection of claim 31 is incorporated. Factor further discloses: *wherein the communication link comprises a local area network.* See section 6 on page 6, bottom of left column to the top of the right column.

In regard to claim 37, the above rejection of claim 31 is incorporated. Factor further discloses: *wherein the commonly referenced memory locations comprise JAVA language fields and the contents or values stored in the commonly referenced memory locations comprise Java field contents or values.* See at least page 1 bottom of right column, e.g. "Java" and "object field."

In regard to claim 39, the above rejection of claim 31 is incorporated. Factor discloses: *wherein application program is written in the JAVA programming language the step of detecting instructions in the unmodified application program which reference the same common memory records comprise searching through the JAVA programming language code and identifying a put static (putstatic) instruction and generating and inserting an alert instruction into the application program for each said putstatic instruction so identified.* See section 3 on page 3, middle of left column, e.g. "updates to an object are detected and propagated."

In regard to claim 40, the above rejection of claim 39 is incorporated. Factor further discloses: *modifying the application program so that during execution of the modified application program upon executing the inserted alert instruction notification, sending the commonly referenced memory record that was re-written or manipulated and may have changed during execution of a code thread with its name tag across the network and receiving the commonly referenced memory record that was re-written or manipulated and may have changed during execution of a code thread with its name tag*

by a different computer. See section 3 on page 3, middle of left column, e.g. "updates to an object are detected and propagated."

In regard to claim 41, the above rejection of claim 31 is incorporated. Factor further discloses: *wherein <network communication> is used for a distributed run time (DRT) communication of the commonly referenced memory record that was re-written or manipulated and may have changed during execution of a code thread with its name tag.* See section 2, page 2, bottom of right column, e.g. "Java socket interface."

In regard to claim 42, the above rejection of claim 31 is incorporated. Factor further discloses: *wherein the updating of all of the commonly referenced memory records that were re-written or manipulated and may have changed during execution of code threads are updated over the Internet.* See section 2, page 2, bottom of right column, e.g. "Internet."

In regard to claim 43, the above rejection of claim 31 is incorporated. Factor further discloses: *wherein the communication link comprises the Internet and all updates to commonly referenced memory locations are performed using Internet network packets through separate distributed runtimes (DRTs) executing on each of the plurality of single computers.* See section 2, page 2, bottom of right column, e.g. "Internet." Further see May for a teaching of separate DRTs as noted above.

In regard to claim 44, the above rejection of claim 31 is incorporated. Factor further discloses: *writing the value from the <network> for the commonly referenced memory record that was re-written or manipulated and may have changed into the memory location of the receiving computer.* See section 3 on page 3, middle of left column, e.g. "updates to an object are detected and propagated."

22. Claim 33 is rejected under 35 U.S.C. 103(a) as being unpatentable over Factor, Scales, and May, and further in view of U.S. Patent 6,101,527 to Lejeune et al. ("Lejeune").

In regard to claim 33, the above rejection of claim 32 is incorporated. Factor, Scales, and May does not expressly disclose: *wherein when the notification and propagation are indirect, the processing of <a thread that has been altered> is only interrupted momentarily before the <altered> thread processing resumes and the other thread which has been notified of the re-written or manipulated commonly referenced memory record then communicates that re-written or manipulated commonly referenced memory record to each of the other single computers so that better utilization of the processing power of various executing threads and gives better scaling with increasing number of single computers when the application program is executed.* However, Lejeune teaches the well-known principle that delegation of tasks to other systems provides greater computing capacity. See column 1 lines 38-43. That is, delegation leads to more computing ability of the system from which the task is delegated. This naturally leads to better utilization of processing power including better scaling. In fact, this

concept is the basis of distributed computing. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Factor's distributed computing with Lejeune's task delegation in order to utillize greater computing capacity as suggested by Lejuene.

23. Claims 25 and 29-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Factor in view of U.S. Patent No. 6,862,608 to Buhlman et al. (hereinafter " Buhlman").

In regard to claim 25, the above rejection of claim 7 is incorporated. Factor does not expressly disclose: *wherein each of the computers operates with the same application program and data and thus all of the plurality of computers have the same application program and data*. However, Buhlman teaches that programs can be operated using the same program and data. See column 1 lines 30-38. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Buhlman's teaching of multiple copies of shared memory with Factor's distributed execution in order to reduce latency as suggested by Buhlman.

In regard to claims 29 and 30, the above rejection of claims 7 and 25 are respectively incorporated. Factor does not expressly disclose: *eliminate clock cycle delays that would otherwise be associated with one or said plurality of computers reading memory physically located in a different one or ones of the plurality of computers formed in a distributed shared memory arrangement*. However, Buhlman

discloses this by way of providing updated data values to each of the distributed processors. Each processor has its own copy of data and therefore does not have to request and wait for a data item from another processor. See column 3 lines 48-62. Thus delays associated with waiting for data from another processor are eliminated. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Buhlman's teaching of multiple copies of shared memory with Factor's distributed execution in order to reduce latency as suggested by Buhlman.

24. Claim 38 is rejected under 35 U.S.C. 103(a) as being unpatentable over Factor, Scales, and May, and further in view of U.S. Patent Application Publication 20030028364 by Chan et al. ("Chan").

In regard to claim 38, the above rejection of claim 31 is incorporated. Factor further discloses: *wherein the commonly referenced memory records comprise JAVA fields.* See at least page 1 bottom of right column, e.g. "Java" and "object field." Factor does not expressly disclose: *the JAVA fields are listed by object and class*. However, May teaches a list of objects. See column 6 lines 23-26. Further, Chan discloses a list of fields in an object class. See paragraph [0003]. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Factor's fields with May and Chan's teaching of field lists in order to easily navigate elements as suggested by Chan.

25. Claims 45-47 are rejected under 35 U.S.C. 103(a) as being unpatentable over May in view of Factor.

In regard to claim 45, May discloses:

In a multiple computer system including a plurality of single computers interconnectable via an Internet or intranet network communications link, a method of loading an original application program onto each of said plurality of single computers, the original application program having original application program code including a plurality of original code threads all written to execute on and reference a single computer having a single processing unit or symmetric multiple processing units and a single local memory with a local memory capacity that is not shared with any other single computer of said plurality of single computers, the system configured to enable simultaneous cooperative execution of said application program by said plurality of single computers, with the original application program being modified to form at least one modified application program with different portions of said modified application program being simultaneously executed within a different independent local processor and a different independent local memory within each different one of the plurality of single computers, said different independent local memory within each said different single computer not forming a distributed shared memory arrangement and being accessible during execution of said application program and said different portions of said application program only by the different portion of the application program actually executing within the different local processing unit or symmetric multiple

processing units of the different computer, See May, column 2 lines 16-28 which generally describes application sharing where distributed computers execute the same program and share the same data. Also see Fig. 1, as well as Fig. 2 which shows an exemplary local processor coupled to local memory.

the method comprising the steps of:

loading the application program onto each different computer of said plurality of single computers, said application program including a reference to a program memory field that may be references by one or more of said plurality of computers during execution of their respective different portion of the application program; and See column 2 lines 32-33, e.g. "manages the adding, deleting, and modifying of the shared data."

...

...an insertion of at least one code thread prior to execution that upon execution by one of said single computers initiates a sequence of events that result in a network packet communication over said Internet or intranet network communications link that contains an identifier of the referenced memory field and the contents or value of that memory field. See Fig. 1 elements 104, 114, and 124 which provides at least one code thread which manages the adding, deleting and modifying of the shared data. See column 2 lines 33-35, e.g. "The OM system also controls the transmitting of modifications to the copy of the shared data to the other computers."

May does not expressly disclose:

modifying the application program on each said different single computer before execution of said different portion of the application program on each said different single computer; and

However, Factor discloses modification of bytecodes to enable distributed execution. See section 4 on page 3, e.g. "Bytecode instrumentation." It would have been obvious to one of ordinary skill in the art at the time the invention was made to use May's application programs with Factor's instrumentation in order to enable an application to be distributed as suggested by Factor.

In regard to claims 46 and 47, May further discloses: *executing said modified application program and generating and communicating said network packet communication over said Internet or intranet network communications link that contains said identifier of the referenced memory field and the contents or value of said referenced memory field.* See column 10 lines 18-23, e.g. "local object handle."

Allowable Subject Matter

26. The following is a statement of reasons for the indication of allowable subject matter:

The examiner indicated that this application would be in condition for allowance if the independent claims 7, 31, and 45 are amended to include the features as indicated in during the examiner initiated interview as shown in the attached appendix. The above features, taken in combination with all remaining features of the independent claim are not taught or suggested by

the prior art of record. The applicant did not agree to amend the independent claims 7, 31, and 45 as indicated by the examiner.

Conclusion

27. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

28. "Orco: A language for parallel programming of distributed systems" by Bal describes distributed computing using logically shared data which does not use physical shared memory. It further describes distributing identical processes to execute using the shared data (see sect56n 3.2 6n page 15 along with Fig. 5 on page 17.

29. U.S. Patent 6,425,016 to Banavar et al. discloses replicated objects (i.e. shared data) for synchronous distributed groupware.

30. Any inquiry concerning this communication or earlier communications from the examiner should be directed to JAMES RUTTEN whose telephone number is (571)272-3703. The examiner can normally be reached on M-F 9:00-5:30.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to JAMES RUTTEN whose telephone number is (571)272-3703. The examiner can normally be reached on M-F 9:00-5:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571)272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/J. Derek Rutten/
Primary Examiner, Art Unit 2192